Delivering Performance, Energy Efficiency and Reliability in a Heterogeneous Exascale System

Corresponding author: Oliver Pell, Maxeler Technologies, oliver@maxeler.com

Introduction and Motivation

Constructing computer systems capable of *exascale* performance clearly poses many significant challenges. At the same time solutions proposed to address some of the obvious problems introduce new research opportunities, in particular, heterogeneity. It is increasingly likely that at a hardware level, exascale systems will utilize heterogeneous compute resources such as high-end CPUs, embedded CPUs, GPUs, dataflow engines (DFEs) and fixed-function compute units. In addition, computing-itself may be distributed between pure compute nodes, storage+compute nodes and interconnect+compute nodes. Exploiting heterogeneity will be critical to achieving the energy efficiency levels necessary to enable exascale computing and an exascale system may contain many different types of heterogeneous elements distributed across a range of locations in the computing system.

In current heterogeneous systems, the heterogeneity is largely invisible to the operating system. Computing devices with different programming and execution models such as GPUs and DFEs are often accessed as I/O devices via library call interfaces. These “devices” must be manually managed by the application programmer, including not just execution of code but also in many cases tasks that are traditionally performed by an OS such as (de-)allocation, load balancing, and context switching. If resources are shared between several OS-running hosts (such as Maxeler’s MPC-X series dataflow appliances, where DFEs are dynamically allocated to CPU processes across the network), issues of contention, fairness and security become further pronounced.

At the same time choreography of data movement is typically left entirely to the user, both within and between compute nodes, something that strains existing programming models and in particular will become more and more problematic as continued transistor scaling makes data movement a larger and larger component of overall energy costs.

At the scale of an exaflop system, manual approaches are likely to be inadequate. Two of the fundamental concerns of exascale systems are energy efficiency and resiliency; both of which will require significant management at the software (operating system) level to deliver required characteristics: for maximum energy efficiency of a computation it may make sense for an operating system or other library to make a decision at runtime over what kind of processing resource to utilize for a particular kernel; for fault management it is vital for software at the cluster level to have some understanding of the different failure modes of different hardware components and the correct response. Even existing systems can have complex fault tolerance behavior – for example in the area of memory errors, CPU servers utilize ECC memory, while GPUs may be configurable to use ECC or not depending on the programmer. An alternative approach which promises flexible scaling of fault tolerance is used in Maxeler systems which provide configurable memory protection on a per data array basis ranging from error detect to single-bit error correct, or higher levels of protection.

Another significant challenge at exascale will be actually delivering this performance to the application. Performance instrumentation requires special attention in heterogeneous environments, since it may make sense to collect different types of data from different resources.

Research Approach

We propose that a research program should address how the software stack below the application level (runtime libraries, operating system etc) takes on management of a heterogeneous computing system. The research program will focus not just on the node but also consider the cluster level, since we anticipate heterogeneity between compute nodes as well as within nodes.

We endeavor to minimize the effort for the programmer to optimize *management* aspects and *performance optimization* aspects of heterogeneous computing resources. This should include resource selection and allocation, arbitration, load balancing, monitoring, energy optimization under synchronization constraints, fault detection and recovery. The operating system or runtime software should enable energy efficient data movement around the system, while providing meaningful abstract interfaces to the programmer.

From the performance perspective, we advocate focusing on two areas: instrumentation and modeling. Instrumentation should be able to capture relevant performance data from the different types of resource in a system, while modeling performance (without execution) will be essential to enable high-level software to make decisions over what resources to utilize for different parts of the application. An end-to-end performance modeling/analysis framework should handle the impact of different types of compute resource, as well as interconnect, storage I/O, etc.

The operating system needs to support adding new types of resources and minimize reliance on underlying hardware features (such as support for virtual memory) which may not be necessary in all kinds of hardware. At the same time, for practical utility the approach should utilize existing standard operating systems (e.g. Linux) rather than developing new operating systems from scratch.

It is imperative to include a strong standardization component, to develop new standards for how operating systems can interact with resources, and how applications can interact with these resources via the OS. The outcome of this work will be an operating system and runtime library environment that is capable of managing all types of compute resource cleanly and effectively, and scaling to an exascale level.

Related Work

- *PTask [1]:* Proposes operating system abstractions to help manage GPUs
- *Nvidia Hyper-Q:* Allows multiple CPU processes to launch jobs on a single GPU
- *Maxeler MPC-X series [2]:* MPC-X allows DFEs (as accelerators) to be dynamically allocated to CPU processes residing on different CPU hosts over Infiniband. The management system permits limited time-slicing of computation on DFEs and allows the programmer to execute a calculation on any DFE that is capable of computing it rather than precisely specifying an allocation.
- *Helios [3]:* Helios is an operating system designed to simplify writing and running code on heterogeneous platforms. It is based upon an intermediate language and final compilation to the target architecture which only suits a limited range of processing technologies.
- *Barrelfish [4]:* An operating system that models the internals of a node as a network of independent cores which communicate by message passing. This approach is also limited to cores which are capable of running a full OS.
- *HARNESS project [5]:* HARNESS is an EU research project that targets enabling the use of heterogeneous computing technologies (compute, networking and storage) for cloud computing through a Platform-as-a-Service (PaaS) environment which abstracts different compute resources from the user while allowing each application to run optimally under some given constraints.

Assessment

- *Challenges addressed:* Energy efficiency, resiliency and fault tolerance, programmability
- *Maturity:* Since heterogeneous processing has only recently come to the fore, software support is lacking at present. However, this research approach should lead to operating systems capable of effectively supporting it in the exascale timeframe. Sufficiently diverse types of processing resource are already available to allow the key characteristics that must be supported to be understood.
- *Uniqueness:* The challenge of managing heterogeneous resources is not unique to exascale systems, however at exascale the problems are more pronounced, for example: automatic fault management is useful in a 100 node cluster, but may be essential to effectively use a 1M node cluster.
- *Novelty:* In existing solutions programmers typically manage compute resources at the application level, with low level operations performed by a device driver. Different kinds of resources are managed independently without any possibility of central coordination.
- *Applicability:* The research results will be applicable to all heterogeneous compute systems.
- *Effort:* Proactive exploration of this approach would involve universities, national labs, supercomputing centers and companies (particularly HPC hardware and software vendors) in perhaps several interlocking research efforts and standardization programs.

References

[1] C. Rossbach et al., PTask: Operating System Abstractions To Manage GPUs as Compute Devices. Proc. SOSP'11, ACM.
[2] Maxeler MPC-X series, dataflow appliance, http://www.maxeler.com/products/mpc-xseries/
[3] E. Nightingale et al. Helios: Heterogeneous Multiprocessing with Satellite Kernels. Proc. SOSP'09, ACM.
[4] A. Baumann et al. The Multikernel: A New OS Architecture for Scalable Multicore Systems. Proc. SOSP'09, ACM.
[5] HARNESS project, http://www.harness-project.eu